

Doy la bienvenida a todos los apasionados del videojuego. **GAME DEVELOPER** ha sido desarrollado por los profesionales más cualificados del panorama nacional para iluminar vuestras mentes ansiosas de conocimiento y guiaros por los intrincados caminos del desarrollo de videojuegos.
Para aquellos más versados en el tema, garantizo que nunca se sabe demasiado de nada y que siempre se puede saber más de todo. Así que si queréis alcanzar un estado de evolución superior a los que os rodean, leer con detenimiento **GAME DEVELOPER** y descubriréis lo que es bueno.
Daremos respuesta a todas vuestras preguntas: programación, grafismo, infografía, composición musical, efectos de sonido, entrevistas con profesionales del sector, ..., no nos hemos olvidado de nada. Incluso se incluyen en nuestro **CD** de portada todos los ejemplos de la revista y el material necesario para que pongáis en práctica, desde el primer instante, todo lo que aquí contamos. Sin ninguna duda **GAME DEVELOPER** es lo que siempre habéis pedido y nadie hasta ahora os había sabido dar.

Total Annihilation dispara las ventas

Cavedog Entertainment ha hecho público que han vendido más de 250.000 copias de Total Annihilation desde su aparición hace pocos días. El aclamado juego de estrategia se vende de momento en tres idiomas (inglés, francés y alemán) así como en 14 países.



Sumario

Formula 1 CE supera los tribunales

Finalmente, Psygnosis podrá vender su esperada ampliación de Formula 1, tras suprimir del *packaging* del juego ciertos logos que no habían permitido su uso. Cuando ya casi lo habían conseguido, una última demanda por parte de la FIA/FOA para volver a detener la producción del juego ha sido desestimada por los tribunales. Por tanto, en breve será comercializado.

Recreativas con corazón Pentium II

Intel, junto con cuatro desarrolladores más, han destapado una nueva estrategia de mercado para dar más salida a los poderosos Pentium II.
Aprovechando la inauguración de la Exposición sobre entretenimiento (AMOA) en Atlanta han presentado la arquitectura abierta para máquinas recreativas basadas en Pentium II.
Los juegos presentados han sido: Ultim@te Race, Plane Crazy y Kick-It. Intel tiene ya el apoyo de Acclaim, Ubi Soft, Gremlin, Microsoft y Sega GameWorks.

SEGA con MMX

Sega Consumer Products anuncia la incorporación de tecnología MMX en todos los lanzamientos previstos de juegos para su línea PC para este otoño/invierno.
Los juegos seguirán siendo compatibles en todo momento con los procesadores Pentium anteriores. Entre los futuros títulos están: Daytona USA Deluxe, segunda entrega del popular Daytona USA. Esta versión incorpora un nuevo circuito además de la optimización para MMX y compatibilidad Force Feedback. Virtua Cop 2 ofrece más enemigos, mayor velocidad y 500 movimientos mejorados. El más esperado es, sin duda, Sega Rally Special Edition. Gracias al set de instrucciones de Intel, MMX, alcanzaremos hasta 30 fps y mejor calidad en los gráficos sin necesidad de tarjeta aceleradora.

Nueva programación para C

Con motivo de la presentación de la nueva programación del canal C:, dedicado al mundo informático de la cadena de pago Canal Plus, fuimos invitados al Hotel Palace, de Madrid, para comprobar in situ lo que la televisión, y en concreto Canal Plus, puede hacer por la informática doméstica y profesional. Para aquellos que no hayan oído hablar de Canal C: habrá que comentar que consiste en uno de los canales temáticos ofertados por Canal Satélite Digital para cubrir las necesidades informáticas de todos los abonados. Desde el mes pasado y, presentando una nueva rejilla de programación, C: ofrece los siguientes programas temáticos:
Lunes Planeta C: una gran página de Internet, según sus creadores. Es un magazine temático en el que se aborda un tema relacionado con la red que nos lleva.
Martes Tangana: los personajes populares, provenientes del mundo de la cultura, los deportes o el espectáculo opinan sobre diversas cuestiones relacionadas con el mundo de los ordenadores.
Miércoles Undercurrents: extraordinaria serie televisiva que explora el impacto de las nuevas tecnologías de la información en la vida cotidiana.
Jueves WaWaWa: visita un cibercafe catalán y conoce a sus inquilinos.
Viernes Cybernet: conoce lo último de lo último en videojuegos para Pc o consolas en este innovador programa. Sólo para especialistas del videojuego.
Sábado Caos: arte y cultura cyber en todas sus expresiones. Monográficos sobre proyectos de arte digital.
Domingo Foro: el director de C:, Virgili Broch, presenta Foro. Programa de debate en el que cuatro invitados aportan ideas, opiniones y puntos de vista sobre las novedades de la actualidad informática.









Destacamos

El CD de portada incluye versiones de evaluación de las siguientes aplicaciones:
- Las fuentes de código de los ejemplos comentados en 3D Manía
- COOL EDIT: editor de samples para generar efectos de sonido de calidad.

Y además... VISTA PRO: generador de mundos virtuales en 3D

Autor: Alberto García-Baquero

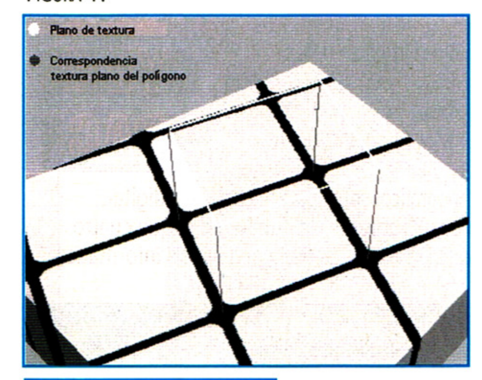
Interpolación afín sobre polígonos

Si bien la utilización de polígonos planos es la base fundamental para la representación de objetos tridimensionales, éstos no son nada nuevos en el mundo de los videojuegos. Así pues, lo que verdaderamente ha revolucionado el uso de las 3D en los videojuegos es el uso de texturas y cada vez más el de diferentes algoritmos de iluminación.

La base del mapeado de texturas es asignar una imagen 2D al plano donde está incluido cada polígono, así cada punto del polígono tiene su coordenada de textura correspondiente en la imagen bidimensional usada como textura. Estas coordenadas de textura dependen de cómo haya sido ajustada la imagen al plano mediante transformaciones afines tales como desplazamiento, rotación y escalado. De esta manera obtendremos las coordenadas que le corresponden a cada vértice del polígono. (Ver figura 1).

Una vez tenemos las coordenadas de textura que corresponden a cada vértice del polígono, interpolaremos estas coordenadas durante la rutina de rastreo de los lados, así, al igual que hacíamos para calcular la intersección con los ejes, calcularemos de forma incremental las coordenadas (u, v) de la textura correspondientes a cada punto. Siguiendo la filosofía de rastreo de polígonos por líneas horizontales que vimos el mes anterior, deberemos calcular los valores de textura por cada eje activo y por cada línea horizontal o scan, como lo llamaremos a partir de ahora. Para ello, se deberá tener en cuenta no sólo la variación de la coordenada y de pantalla sino también la consecuente variación de la x. Para este propósito necesitaremos calcular las derivadas/pendientes/deltas/incrementos de cada componente de la textura en función de la variación de la x e y en pantalla. Ahora procederemos a explicar cómo se hallan estas derivadas.

FIGURA 1.



INTERPOLACION AFIN A TRAVES DE UNA SUPERFICIE BIDIMENSIONAL

Partiendo de que los valores que queremos interpolar son afines a las coordenadas de pantalla (*x*,*y*) tenemos que:

$$Nxy = dNx \cdot x + dNy \cdot y + No$$

En este caso, *N* es la propiedad del punto que deseamos interpolar linealmente en el espacio de la pantalla, que según el caso podría ser la coordenada *u* o *v* de la textura, el valor de iluminación gouraud, etc. *dNx* y *dNy* son las derivadas de *N* en función de la *x* e *y* de pantalla respectivamente y *No* el valor de *N* en el punto tomado como origen.

Para obtener el valor de las derivadas necesitaremos al menos tres puntos; tomando uno de ellos como el origen, obtendremos el siguiente sistema de ecuaciones:

1) Tomamos tres puntos cualquiera del polígono:

$N0 = dNx \cdot X0 + dNy \cdot Y0 + No$
$N1 = dNx \cdot X1 + dNy \cdot Y1 + No$
$N2 = dNx \cdot X2 + dNy \cdot Y2 + No$

2) Tratamos uno de los tres puntos como el origen:

$dN1 = N1\text{-}N0,\ dX1 = X1\text{-}X0,\ dY1 = Y1\text{-}Y0$
$dN2 = N2\text{-}N0,\ dX2 = X2\text{-}X0,\ dY2 = Y2\text{-}Y0$
$dN1 = dNx \cdot dX1 + dNy \cdot dY1$
$dN2 = dNx \cdot dX2 + dNy \cdot dY2$

3) Despejamos *dNx*, e igualamos:

$(dN1 - dNy \cdot dY1) / dX1 = (dN2 - dNy \cdot dY2) / dX2$

$(dN1 \cdot dX2) - (dY1 \cdot dX2) \cdot dNy = (dN2 \cdot dX1) - (dY2 \cdot dX1) \cdot dNy$

4) Hallamos *dNy* y *dNx*:

$dNy = ((dN1 \cdot dX2) - (dN2 \cdot dX1)) / ((dY1 \cdot dX2) - (dY2 \cdot dX1))$
$dNx = ((dN1 \cdot dY2) - (dN2 \cdot dY1)) / ((dX1 \cdot dY2) - (dX2 \cdot dY1))$

$InvdY = 1.0 / (dX2 \cdot dY1) - (dX1 \cdot dY2)$
$InvdX = 1.0 / (dX1 \cdot dY2) - (dX2 \cdot dY1)$
$InvdX = - InvdY$

Concretamente, *InvdX* e *InvdY* son característicos del polígono y constantes para el cálculo de todos los deltas de las propiedades del polígono. Como se puede ver, *InvdX* e *InvdY* son opuestos, con lo que sólo es necesario calcular el valor de uno de ellos y asignar el opuesto al otro inverso. Esto supone que mediante este método, para calcular todos los deltas de las propiedades constantes de un polígono, sólo es necesaria una división.

5) Resultado:

$dNx = ((dN1 \cdot dY2) - (dN2 \cdot dY1)) \cdot InvdX$
$dNy = ((dN1 \cdot dX2) - (dN2 \cdot dX1)) \cdot InvdY$

En el caso de que el polígono esté formado por más de tres puntos y alguno de éstos no cumpliese que $Nxy = dNx \cdot x + dNy \cdot y + No$, *N* (la propiedad siendo interpolada) no tiene un delta constante en función de las variaciones de *x* e *y* en pantalla. Esto puede ocurrir debido a diversas razones que veremos más adelante. Para estos casos, existen varias alternativas como triangulizar los polígonos, o calcular la correspondiente pendiente de cada propiedad a interpolar por cada eje y por cada línea horizontal, que teniendo en cuenta que cada cálculo dependiente supone al menos una división, resulta considerable el costo del proceso; a pesar de ello, este método es el más

extendido y usado por la mayoría de programadores, sin embargo, el que un polígono carezca de deltas constantes, supone que líneas paralelas en la textura original no lo serán en su aplicación sobre el polígono, siendo rotacionalmente variables. Esto quiere decir que se deforman de manera distinta según su orientación, por lo cual este hecho no debe ser un impedimento, sino que deberemos evitarlo en la mayoría de los casos, buscando otros métodos de interpolación que veremos más adelante.
Sin embargo, con el actual método tan solo es necesaria una división para calcular todos los deltas constantes de los valores que se deseen interpolar a lo largo del polígono, sin importar el número de ejes, vértices o tener en cuenta la concavidad/convexidad del polígono.
Una vez sabemos calcular los incrementos constantes necesarios, describiremos la metodología a seguir durante la interpolación de un valor que es afín a las coordenadas de pantalla, esto quiere decir que tiene incremento constante respecto a la variación de las coordenadas de pantalla:

Durante la inicialización de los valores del polígono realizamos los siguientes cálculos:

- Calculamos *Invdx.*
- Calculamos los deltas constantes respecto a las coordenadas de pantalla (*dNx y dNy*), de las propiedades que deseamos interpolar (coordenadas de textura, iluminación gouraud, etc.).

Durante la inicialización de los valores de cada eje:

- Por cada eje, a excepción de los ejes de orden par en el caso de polígonos no complejos, precalculamos: *dN = dNx · DeltaX + dNy*, que será el incremento mínimo de *N* por cada avance de línea sobre el eje. Mínimo porque, en caso de acarreo de la parte fraccionaria de *X* y que se produzca un avance extra de X, como se vio en el artículo del mes pasado, se producirá por lo tanto una variación extra de *dN* que, al ser debida al incremento de 1 en la coordenada *X* de pantalla, será igual a *dNx*.

Por cada avance de línea:

- Actualizaremos las propiedades interpoladas en los ejes activos.

Para calcular el nuevo valor de la propiedad por cada eje se hará de forma incremental, haciendo uso del delta mínimo correspondiente a cada avance de línea y que ha sido precalculado por cada eje.
En el caso de que se haya producido un avance de la *X* por acarreo de su parte fraccionaria, se habrá de hacer un incremento extra de *dNx*.
Para los polígonos no complejos (aquellos cuyos ejes no se intersectan) que son la mayoría, sólo será necesario en los ejes impares, que son los que inician los segmentos, ya que los valores finales no son necesarios pues éstos se calculan a partir de los anteriores.

Por cada punto durante el dibujado de cada línea horizontal:

- Por cada línea calcularemos el valor de *N* para todos los puntos de izquierda a derecha; para ello tomaremos como valor inicial el del eje izquierdo y, según avancemos la coordenada *x* de pantalla, actualizaremos la propiedad *N* sumándole su derivada en función de *x (dNx).*

APLICACION DE UNA TEXTURA A UN POLIGONO

Una vez que sabemos interpolar, de manera afín a las coordenadas de pantalla, una propiedad *N* a lo largo de un polígono, para aplicar una textura a un polígono, únicamente debemos interpolar las correspondientes coordenadas *u, v* de la textura, esto es, a cada punto del polígono le corresponde un punto en la imagen usada como textura cuyas coordenadas son (*u, v*). Así pues, las modificaciones a hacer respecto a la anterior rutina de pintar polígonos convexos planos son básicamente las siguientes:

- Durante la inicialización del polígono calcularemos los deltas constantes de (*u, v*) tal y como se explicó anteriormente y puede verse en el Listado 1.
- En la inicialización de los ejes de la izquierda, y sólo los de la izquierda, puesto que dibujamos de izquierda a derecha y sólo hacen faltas los valores de comienzo, calculándose el resto incrementalmente, inicializaremos también los correspondientes valores de (*u,v*), así como sus deltas mínimos *dU y dV*. Para ello, procederemos tal y como muestra el Listado 2.
- Por cada avance de línea, al igual que actualizamos el valor de la coordenada *x* de pantalla, actualizaremos los valores de las coordenadas *u,v* de textura. Así pues, le sumaremos su correspondiente delta mínimo, calculado durante la inicialización de cada eje, y, en caso de acarreo, incrementaremos las coordenadas con su correspondiente delta en función del avance de *x* en pantallas; esto es *dUx* y *dVx*, valores globales para todo el polígono. Esta tarea, en el caso de los polígonos convexos, sólo es necesaria llevarla a cabo para los ejes del lado izquierdo. Para ellos utilizaremos la nueva función modificada a partir de *StepX, StepXUV* que puede verse en el Listado 3.
- Por último, la más importante de todas, lo que se le suele denominar el bucle interior de dibujado (*inner loop*) y, por lo tanto, el que más veces es ejecutado. En este bucle dibujaremos las líneas

LISTADO 1

```
// Calculamos invdx y los deltas
// constantes de U y V
  Vertex  *P0 = P->Vertices[0];
  Vertex  *P1 = P->Vertices[1];
  Vertex  *P2 = P->Vertices[2];
  float  dX1 = P0->x - P1->x;
  float  dX2 = P0->x - P2->x;
  float  dY1 = P0->y - P1->y;
  float  dY2 = P0->y - P2->y;

  InvdX = 1.0 / (dY2*dX1 - dX2*dY1);
  InvdY = -InvdX;

  float  dU1 = P0->u  - P1->u;
  float  dU2 = P0->u  - P2->u;
  dUx = (dY2*dU1 - dU2*dY1) * InvdX;
  dUy = (dX2*dU1 - dU2*dX1) * InvdY;

  float  dV1 = P0->v  - P1->v;
  float  dV2 = P0->v  - P2->v;
  dVx = (dY2*dV1 - dV2*dY1) * InvdX;
  dVy = (dX2*dV1 - dV2*dX1) * InvdY;
```

LISTADO 2

```
Eje2D::CalcDeltaXUV()
{
// Inicializamos los valores generales
// por cada eje.
CalcDeltaX();

// Valores de inicio de (u,v)
U = P->u;
V = P->v;

// Calcula deltas mínimos de las
// coord. De textura (u,v)
dU = dUx*XStep + dUy;
dV = dVx*XStep + dVy;
}
```

LISTADO 3

```
// Avanza un span de un eje,
// coord. x y de texturas
Eje2D::StepXUV()
{
   U     += dU;
   V     += dV;
   X     += XStep;
   XError += Numerador;
   if( XError >= Denominador )
    {
    X++;
    U     += dUx;
    V     += dVx;
    XError -= Denominador;
    }
}
```

horizontales en las que hemos dividido el polígono y, por ello, todos los puntos del mismo. Por este motivo es la parte más importante de una rutina rápida de texturas, el corazón de la misma. Es aquí donde se hace mayor consumo de tiempo y procesador y, debido a esto, donde más se debe optimizar el código y algoritmo, tema al que dedicaremos próximos artículos. Lo primero que haremos antes de empezar a dibujar el primer punto será inicializar los distintos valores necesarios. Estos valores serán las coordenadas iniciales de (*u* ,*v*) que se tomarán del valor actual de las mismas en el eje izquierdo, desde donde se comienza el dibujado de la línea horizontal o scan-line. Tras asignar los valores iniciales comienza un bucle desde la coordenada *x* del eje izquierdo hasta el eje derecho. En el interior de este bucle extraeremos el color correspondiente de la textura y lo pintaremos en la pantalla; avanzaremos la *x* y calcularemos, por cada avance de ésta, las correspondientes coordenadas de (*u,v*); para ello bastará con sumarles a las (*u,v*) actuales su correspondiente delta constante en función del avance de *x* que calculamos al principio de la función: *dUx y dVx*.

Todo este proceso viene a sustituir al anterior dibujado de líneas horizontales planas que se limitaba a rellenar todos los puntos de la línea con el mismo valor. La implementación básica del mismo puede verse en el Listado 4.

Sin embargo, este ejemplo es demasiado lento ya que la multiplicación por línea y las conversiones de flotante a entero toman bastante tiempo de procesador, por ello, a pesar de que trataremos en próximos artículos de manera profunda la optimización de este código, implementaremos dos mejoras básicas para hacerlo mas rápido. Una de estas mejoras será utilizar números enteros con coma fija en lugar de flotantes; estos, que también se tratarán en próximos artículos, se basan principalmente en utilizar un numero fijo de bits para la parte entera y otro número fijo para la parte decimal. En este ejemplo utilizaremos 16.16, que se traduce en que los dos bytes de mayor peso contendrán la parte entera, y los de menor peso la parte decimal. Para pasar de flotantes a enteros con coma fija 16.16 deberemos multiplicar el número flotante por: 1.0 en coma fija 16.16 = 0x1000 = 65536.

Ej: coma_fija = flotante * 0x1000.

Posteriormente, cuando deseemos tomar la parte entera de un número en coma fija 16.16, lo desplazaremos a la derecha 16 bits.

Ej: entero = coma_fija >> 16.

Entonces, durante la inicialización de valores, convertiremos a coma fija los actuales valores de *u,v* y los deltas constantes de *u,v* en función de la variación de la *x* de pantalla. Y en el bucle interno, en lugar de convertir de flotantes a entero, haremos un simple desplazamiento de bits.

Otra optimización básica que llevaremos a cabo será el uso de textura con un ancho cuyo valor sea potencia de 2 para, de este modo, sustituir la costosa multiplicación por un simple desplazamiento de bits; en el ejemplo utilizamos un ancho de 256 = 2^8 por lo que, para eludir la multiplicación, sólo debemos hacer un desplazamiento de 8 bits a la izquierda.

Con estas mejoras básicas, quedando el código tal y como se puede ver el listado 5, se consigue una velocidad razonable.

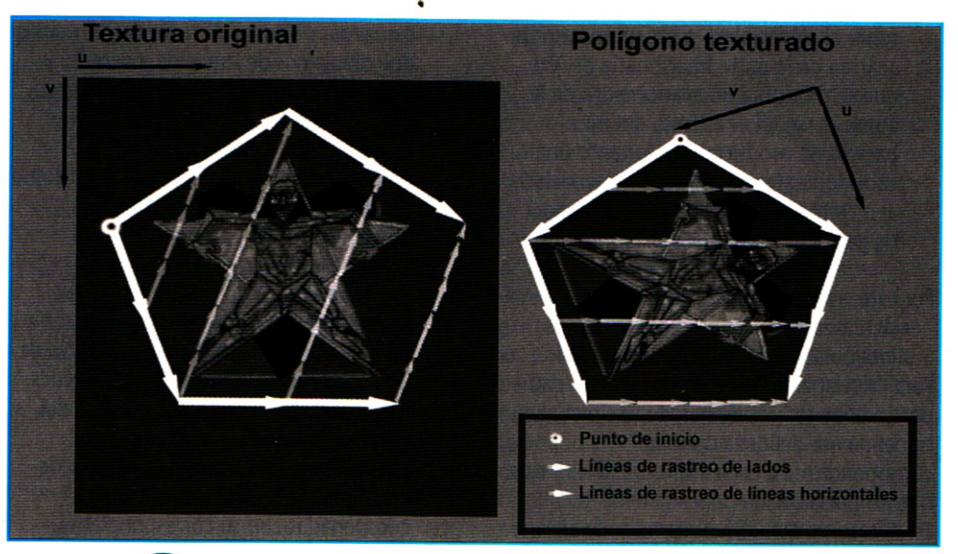
APLICACION DE ILUMINACION GOURAUD

La iluminación de polígono mediante gouraud se basa, principalmente, en interpolar linealmente la iluminación de los distintos vértices del polígono. El cálculo de la iluminación de cada vértice del polígono no corresponde a este artículo, y depende de valores como la orientación del polígono en el espacio, las fuentes de luces o focos que afectan a cada vértice, polígonos vecinos, etc. Por todos estos parámetros, en este algoritmo de iluminación, al igual que en casi todos, la iluminación de cada vértice del polígono es independiente, de ahí que suelan

LISTADO 4

```
/// Inicializamos valores
//
int         X = EIzq.X;
float       U = EIzq.U;
float       V = EIzq.V;
while(X < EDer.X)
{
  // Lee punto (u,v) de la textura y lo pinta.
int OffS = ((int)U) +
((int)V)*AnchoTextura);
  *(Out++) = TEXTURA[OffS];
  // Avanza x, actualiza valores de u,v
  X++;
  U+=du;
  V+=dv;
}
```

FIGURA 2.



LISTADO 6

```
// Cálculo de pendiente del lado con flotantes, para
// tener precisión sub-pixel.
float DifX = NextP->x - P->x;
float DifY = NextP->y - P->y;
float Delta = DifX / DifY;
Denominador   = 1.0f;
XStep = (int)( Delta );
if( DifX > 0 )   Numerador = Delta - XStep;
else {
    Numerador   = XStep - Delta;
    if(Numerador) {
        XStep—;
        Numerador = Denominador - Numerador;
    }
}

// Calcula coordenadas iniciales y efectua ajuste
// subpixel para la X
X = (int)P->x;
XPrestep = X - P->x;
YPrestep = (((int)P->y) - P->y);

// Ajusta subpixel para los ejes
XError   = (Delta * YPrestep) - XPrestep;

// Ajusta subpixel para coordenadas de texturas
// tambien llamado subtexel
U = P->u + dUy*YPrestep + dUx*XPrestep;
V = P->v + dVy*YPrestep + dVx*XPrestep;
```

carecer de deltas constantes los polígonos con más de tres lados. Para estos casos dedicaremos próximos artículos. Sin embargo, para casos de polígonos con deltas constantes para interpolar su iluminación, se procederá tal y como se ha explicado anteriormente y se ha hecho con la interpolación de coordenadas de textura,

LISTADO 5

```
// Inicializamos los valores
// y convertimos de float —> 16.16 coma fija
int Count = EDer.X-EIzq.X;
int U = EIzq.U*0x10000, du = dUx
*0x10000;
int V = EIzq.V*0x10000, dv = dVx
*0x10000;

while(Count— >0)
{
    int OffS = (U>>16) | ((V>>16)<<8);
    *(Out++) = TEXTURA[OffS];
    U+=du;
    V+=dv;
}
```

pero en este caso concreto, con sólo una propiedad a interpolar, la iluminación. Así, siguiendo los métodos ya explicados, se buscará la iluminación correspondiente a cada punto del polígono. Esta iluminación puede ser calculada de diferentes maneras, pudiendo tratarse de iluminación por intensidad o luz blanca, en la que solamente se maneja un valor, la cantidad de luz, o tratándose las luces independientemente por cada componente RGB(rojo, verde, azul), pudiéndose representar así luces de distintos colores.

AÑADIENDO PRECISION: SUBPIXEL

Sin embargo, aunque hallamos seguido todos los pasos ya comentados, y tal como podemos observar en el segundo ejemplo incluido en el CD de la revista, el polígono no gira ni se desplaza suavemente, sino que se mueve a saltos, punto a punto. Este fenómeno puede verse en juegos como Tomb Raider, donde si bien no es apreciable a primera vista y no constituye un gran defecto, sí que sería bastante valiosa la inclusión del mismo. Un buen ejemplo entre el 'tener o no tener' subpíxel podría ser el comparar el movimiento de los polígonos del Tomb Raider con el efecto conseguido al pasar de fase en Quake, donde se puede ver una vista panorámica de la fase con un sutil balanceo de cámara que confiere una gran sensación de solidez y realismo. Este fenómeno se debe a que, al pintar en pantalla, ignoramos la parte no entera de las coordenadas de pantalla, con lo que se cometen errores de hasta casi 1 punto de precisión, tanto en los valores interpolados a lo largo del polígono como a los lados del polígono. Para evitar que estos fallos ocurran, deberemos ajustar los valores interpolados según el desfase de las coordenadas del punto original con las coordenadas truncadas de pantalla. Para ello, calcularemos la distancia de las coordenadas enteras truncadas de pantalla a las coordenadas flotantes originales. Con este desfase calculado y los deltas en función de la variación de *x,y* en pantalla ajustaremos el valor de la propiedad al que tendría en las coordenadas truncadas de la siguiente manera:

DifX = ((int)x)-x
DifY = ((int)y)-y
N = No + dNx · DifX + dNy · DifY

Esta corrección, gracias a que mediante el uso de DDA controlamos cada variación entera de las coordenadas *x,y*, únicamente debemos efectuarla durante la inicialización de los valores de los ejes que, en el caso de los polígonos convexos, sólo será necesario con los ejes del lado izquierdo del polígono el ajustar subpíxel para las propiedades interpoladas, aunque el ajuste subpíxel de la *x* deberá hacerse para todos los ejes. Si no fuera porque controlamos cada avance entero de las coordenadas de pantalla, deberíamos efectuar una corrección por cada línea horizontal, con lo que el ahorro del método aquí expuesto es considerable. Sin embargo, para conseguir la precisión deseada, deberemos utilizar flotantes en lugar de enteros para calcular la pendiente. Al igual que se corrigen las propiedades interpoladas a lo largo del polígono, deberemos ajustar el error acumulado en la *X*. Para ello calcularemos el desfase de la coordenada *y*, y su pendiente en función de ésta. Con estos valores, resultará que el error acumulado inicial será igual al desfase de la *x* más el ajuste en función del desfase de la *y*.
Todas estas operaciones se pueden ver aplicadas en el Listado 6 y en el código fuente del ejemplo 3 incluido en el CD de la revista.
La implementación del ajuste subpíxel es un requisito imprescindible, puesto aunque los resultados obtenidos sin él son aceptables, sí que supone una mejora de precisión bastante considerable notándose, sobre todo, en ligeros movimientos de cámara siendo uno de los 'pequeños detalles' que hacen que un juego sea de calidad. Esto lo podréis notar si os fijáis atentamente en las diferencias existentes entre el ejemplo 2 y el ejemplo 3 incluidos en el CD.
Para el próximo número, siguiendo con el tema de la aplicación de texturas y la precisión, trataremos distintos métodos de interpolación para pintar texturas aplicadas a polígonos en 3D, para lo que introduciremos nuevos términos básicos como la proyección de un punto 3D en pantalla, ángulo de visión, etc., para poder introducirnos en la aplicación de texturas con corrección de perspectiva y tratar los diferentes métodos existentes.

Alberto García-Baquero
Wisefox@cyberdude.com
Albertogb@jet.es
Wisefox@jet.es

Autor: Leticia Krahe

Prográmate tu propia página web

Muchas veces os habréis preguntado cómo se programarán esas páginas tan bonitas que están en la red; bueno, pues en esta sección pretendemos que vosotros mismos podáis crearos de una manera sencilla y divertida vuestra WEB PERSONAL.

En primer lugar hay que decir que aprenderemos a crear web´s mediante una utilidad de libre distribución, denominada Hot Dog, que es bastante sencilla de manejar.
En este capítulo vamos a ver cómo introducir texto e imágenes, los diferentes títulos que podremos crear, las distintas fuentes de letras y, por último, qué formato de imágenes se pueden meter y cómo introducirlas dentro de la Web.

PRIMER CONTACTO

Una vez instalado, lo arrancaremos; entonces aparece una pantalla con cuatro opciones, usar HotDog ahora, contar algo sobre HotDog, contar algo sobre html y, para finalizar, un tutorial sobre html. Los curiosos o los que quieran aprender más rápido podrán pulsar sobre cualquiera de las tres últimas opciones, pero aquellos que deseen seguir paso a paso esta guía deberán pulsar sobre la primera opción, usar HotDog.



Cuadro de inicio de HotDog.

Una vez pulsada esta opción, ya podremos empezar a crear nuestra propia página. Hot Dog nos escribe él sólo, todo lo necesario para que funcione la página, es decir, al comenzar aparece lo siguiente:

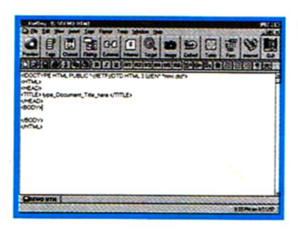


Figura 1.

Estas son las etiquetas necesarias para empezar a funcionar, la segunda (<HTML>) indica que es un fichero HTML y la última (/HTML) es que es el final del fichero HTML. <HEAD> y </HEAD> indican comienzo y fin de cabecera, respectivamente. Por su parte, <TITLE> indica el comienzo del título que aparece en la parte superior junto a Internet Explorer o Netscape, etc., y </TITLE> el final del título. Donde Hot Dog escribe type_Document_Title_here deberemos sustituirlo por el título que queramos, por ejemplo, "demostración de HTML". Por último, <BODY> y </BODY> indican comienzo y final del cuerpo; esta parte es la más importante, ya que aquí es donde escribiremos la página web.

CABERECERAS O TITULOS

Introducir cabeceras o títulos es algo bastante importante dentro de las páginas web´s más sencillas, aunque en lugar de poner una cabecera se suele colocar una imagen que venga a decir lo mismo, pero para aquellos que se inicien en este campo resulta, por lo menos, algo interesante.
Hay seis tipos de títulos o cabeceras, y dos formas de realizarlo. La primera que os vamos a explicar es la más sencilla: situaremos el cursor en el espacio situado entre <BODY> y </BODY>, y pulsaremos sobre H1, H2, H3, H4, H5 ó H6, según el tipo de cabecera que queramos y escribir entre <H(número)> y </H(número)> el título que le queramos poner a nuestra página web, por ejemplo "MI PÁGINA PERSONAL". La segunda manera es teclear directámente la indicación de título, por ejemplo <H2>.
A continuación aparece el formato que tienen estas cabeceras.

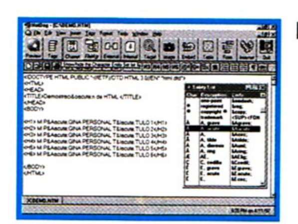


Figura 2.

Tal vez os preguntaréis cómo este código sale después en el Explorer o en cualquier otro programa de Internet; bueno, esto es bastante sencillo. En primer lugar debéis salvar este documento HTML, para ello, seleccionar *File* y *Save As...*, entonces aparecerá una pantalla, en la cual debéis introducir el nombre del documento, en nuestro caso Demo.htm. Después, debéis pulsar la tecla F5 ó seleccionar *File* y Preview Document o, simplemente, pulsar sobre el icono *PREVIEW* entonces os aparecerá lo siguiente:

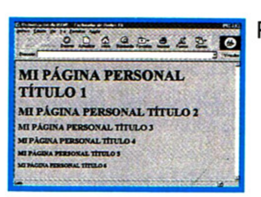


Figura 3.

Para centrar un título o texto sólo tenéis que poner el título o el texto que queráis centrar entre las etiquetas <CENTER> y </CENTER>; éstas las podéis escribir bien tecleándolas directamente, bien pulsando sobre el icono de *CENTRADO*.

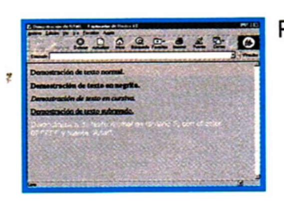
CUIDADO CON ESOS CARACTERES

Un aspecto que debéis tener en cuenta es que hay caracteres que no se pueden escribir tal y como son; por ejemplo, las letras acentuadas, es decir, la "á" no se escribe así, si no á.
Para escribir estos caracteres directamente en el HotDog debéis pulsar sobre el icono de *CHARSET* y aparecerá la ventana, *Entity List*, en la que si pulsáis con doble clic sobre el carácter que queréis poner. Este editor lo pondrá sólo, pero ¡cuidado! debéis tener el cursor bien situado, es decir, justo donde queráis que vaya ese carácter. En la Figura 3 podéis ver esta ventana.

INTRODUCIR TEXTO

Este apartado es fundamental, ya que todas las páginas tienen texto. Para introducir texto debéis poner <P> o pulsar sobre el icono de párrafo, teniendo en cuenta que esta etiqueta no tiene la correspondiente de fin de párrafo, es decir, no existe la etiqueta </P>. La etiqueta <P>, seguida de texto, indica que es un párrafo, pero si está seguida de otra etiqueta muestra una línea en blanco.
Si no os gusta en demasía el tipo de letra que se pone por defecto, no os preocupéis, ya que ésta se puede cambiar. Para llevar a cabo estos cambios basta con poner:
<FONT SIZE=3 COLOR=#0FFFFF FACE=

"Arial">, a continuación escribir el texto y, por último, poner la etiqueta </FONT>.
SIZE indica el tamaño de la letra y está comprendido entre el 1 y el 7, donde el 1 es el tipo de letra más pequeño y el 7 el más grande. COLOR indica la tonalidad en el que se va a escribir la letra; este valor se debe poner en hexadecimal, por ejemplo, el número #0FFFFF corresponde a azul celeste. Por último, FACE hace referencia al tipo de fuente que se quiera utilizar. Una cosa a tener en consideración es que el lector de nuestra página a lo mejor no tiene el tipo de letra Arial, por lo que, en este caso, la letra se mostrara en el fuente estándar "Courier".
A continuación tenéis una demostración de los textos, en el cuadro 1.
Ejecutando éste código veríamos lo siguiente:


FIGURA 4.

¿IMAGENES?

Como reza un dicho, *"Una imagen vale más que mil palabras"*, y en este campo sucede lo mismo. Las imágenes, por decirlo de alguna manera, dan "vidilla" a nuestra página personal. Podemos tener dentro de la página sólo imágenes, sólo texto o texto con imágenes, siendo esta última opción la que mayor vistosidad da a nuestra página.
En HTML sólo se pueden meter dos tipos o formatos de imágenes, GIF'S y JPG'S, es decir, cualquier otro formato no se puede introducir dentro de una web, por lo menos utilizando HotDog.
En primer lugar veremos cómo se introducen imágenes, y en segundo lugar cómo se juntan imágenes con texto.

IMAGENES SOLAS

Para incluir imágenes debéis hacer lo siguiente: pulsar sobre el icono *IMAGE*, entonces aparecerá una ventana. En *Image File* debéis introducir el nombre de la imagen y en qué directorio se encuentra. Una vez introducido del nombre debéis pulsar Aceptar. Entonces HotDog os añadirá la siguiente línea en el documento, que debe estar situada dentro de <BODY>.

<IMG SRC="demo.jpg">

Ejecutando este código os aparecerá la página con la imagen.

IMAGEN CON TEXTO

Para juntar imágenes con texto hay que saber bien lo que se quiere; por ejemplo se puede introducir una imagen como título y debajo de ésta texto. También se podría intercalar texto e imágenes, con lo que habría que saber si la imagen la queremos alinear con el texto en la parte superior, en la inferior o en el centro.
Para ponerla como título y centrada y con texto debajo deberemos escribir:

```
<CENTER>
<IMG SRC="demo.jpg">
</CENTER>
<FONT SIZE=4>
<P>
<P> Ejemplo de foto como
t&iacute;tulo y debajo texto normal.
```

Para ponerla alineada con el texto, pulsaremos sobre cualquiera de los tres iconos que alinean una imagen. (* ICONOS.TIF *) Otra opción sería alinearla a la derecha ó a la izquierda; para esto deberemos escribir ALIGN=LEFT ó RIGHT, con lo que conseguiremos que todo el texto se escriba junto a la foto. Si escogemos alinear la fotografía con el texto a la izquierda, el código quedará de la siguiente manera:

```
<IMG SRC="demo.jpg"
ALIGN=LEFT> Ejemplo de foto
acompa&ntilde;da de texto, el texto está
acompa&ntildeando a la imagen.
```

UN EJEMPLO A SEGUIR

En el cuadro 2 tenéis el código de un ejemplo para que podáis empezar a practicar Este ejemplo consiste en un gráfico como título y algunas de las cosas que pondríamos en nuestra propia página. Por ejemplo, a qué nos dedicamos, nuestro hobbies, etc... Ahora pulsaremos sobre el icono de PREVIEW y nos saldrá nuestra esperada página personal.



UN EJEMPLO BASICO.

Por último, solamente deciros que esperamos que, por lo menos, os lo paséis bien montando vuestra propia página; en el siguiente número seguiremos contando cosas sobre HTML, por ejemplo, como crear enlaces para tener varias páginas y muchas cosas más.

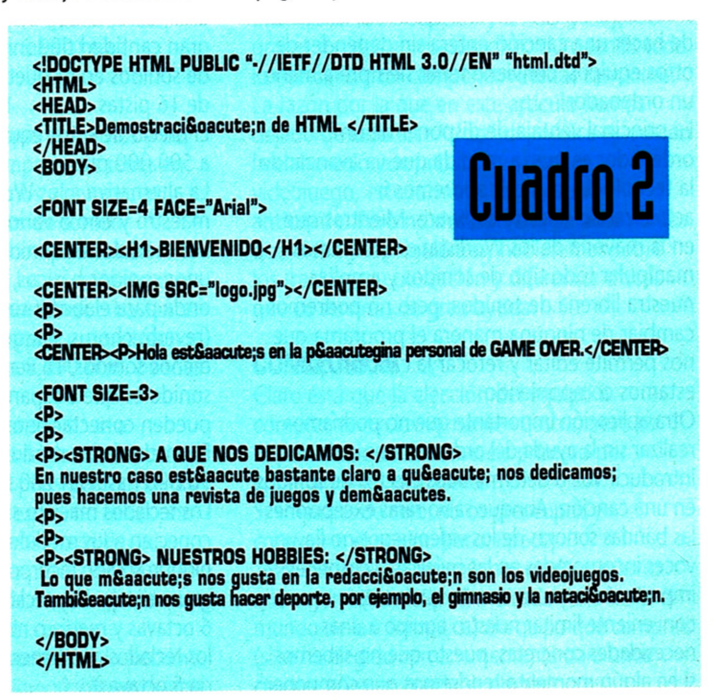
Cuadro 1

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.0//EN"
"html.dtd">
<HTML>
<HEAD>
<TITLE>Demostraci&oacute;n de HTML </TITLE>
</HEAD>
<BODY>

<P>
Demostraci&oacute;n de texto normal.
<P>
<STRONG>Demostraci&oacute;n de texto en negrita.</STRONG>
<P>
<EM>Demostraci&oacute;n de texto en cursiva.</EM>
<P>
<U>Demostraci&oacute;n de texto subrayado.</U>

<FONT SIZE=3 COLOR=#0FFFFF FACE="Arial">
<P>Demostraci&oacute;n de texto normal de tama&ntilde;o 3,
con el color 0FFFFF y fuente "Arial".
</FONT>

</BODY>
</HTML>
```


Cuadro 2

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.0//EN" "html.dtd">
<HTML>
<HEAD>
<TITLE>Demostraci&oacute;n de HTML </TITLE>
</HEAD>
<BODY>

<FONT SIZE=4 FACE="Arial">

<CENTER><H1>BIENVENIDO</H1></CENTER>

<CENTER><IMG SRC="logo.jpg"></CENTER>
<P>
<P>
<CENTER><P>Hola est&aacute;s en la p&aacutegina personal de GAME OVER.</CENTER>

<FONT SIZE=3>
<P>
<P>
<P><STRONG> A QUE NOS DEDICAMOS: </STRONG>
En nuestro caso est&aacute bastante claro a qu&eacute; nos dedicamos;
pues hacemos una revista de juegos y dem&aacutes.
<P>
<P>
<P><STRONG> NUESTROS HOBBIES: </STRONG>
 Lo que m&aacute;s nos gusta en la redacci&oacute;n son los videojuegos.
Tambi&eacute;n nos gusta hacer deporte, por ejemplo, el gimnasio y la nataci&oacute;n.

</BODY>
</HTML>
```

Autor: Víctor Segura

Estudios caseros

¿Alguna vez has querido tener un estudio de grabación? ¿No sabes cómo aprovechar al máximo tu equipo para poder hacer tu propio estudio? Este mes te daremos una serie de consejos muy útiles para que puedas tener en tu propia casa un verdadero estudio semiprofesional.

Antes de adentrarnos en la elaboración de la banda sonora de un videojuego, conviene revisar el equipo que vamos a tener a nuestra disposición, así como su funcionalidad y capacidad para satisfacer nuestras necesidades. Tenemos que tener en cuenta que el hecho de tener un equipo con muchos componentes no implica una mayor calidad o un mejor resultado final en la creación de la banda sonora. Es más importante tener un equipo medio y aprovecharlo al máximo, que tener un sinfín de teclados, procesadores de sonido etc., de los que sólo aprovechamos un 25% de sus características.

EL ORDENADOR: NUESTRO MEJOR ALIADO

La base de nuestro estudio va a ser sin lugar a dudas el ordenador. Aunque los equipos denominados Workstation (teclados que incluyen sonidos, herramientas para manipular estos sonidos y un secuenciador) se valen por sí solos para poder llevar a cabo la ardua tarea de hacer una canción entera sin depender de otros equipos, conviene tener siempre a mano un ordenador .

La principal ventaja de disponer de un ordenador es que, a medida que va avanzando la tecnología musical, podremos ir actualizando nuestro software. Mientras que en la mayoría de los Workstation podremos manipular todo tipo de sonidos y ampliar nuestra librería de sonidos, pero no podremos cambiar de ninguna manera el programa que nos permite editar y retocar la canción que estamos componiendo.

Otra aplicación importante que no podríamos realizar sin la ayuda del ordenador es la de introducir voz o determinados efectos de sonidos en una canción. Aunque salvo raras excepciones las bandas sonoras de los videojuegos no llevan voces incorporadas en las músicas, siempre es importante tener en cuenta que no es conveniente limitar nuestro equipo a unas necesidades concretas, puesto que no sabemos si en algún momento tendremos que componer determinadas canciones con voz o efectos de sonidos incorporados.

WORKSTATION, TECLADOS MAESTROS Y MÓDULOS

Como hemos dicho anteriormente un Workstation es, como su propio nombre indica, una estación de trabajo. Con un equipo de estas características, se puede elaborar una canción sin tener que depender de otros aparatos. Ni que decir tiene que con un Workstation y un ordenador, se pueden hacer cosas realmente importantes, de hecho, es todo lo que se necesita para empezar a componer.

En realidad un Workstation es un módulo de sonidos, un teclado maestro y un pequeño ordenador todo incorporado en un mismo equipo. El problema que nos encontramos en un Workstation es que como tienen "todo" incorporado en un solo aparato, el precio es elevado. Normalmente son equipos que tienen una polifonía de 32 o 64 notas, 5 octavas, una gran cantidad de sonidos (incluyendo librerías de sonidos en disquetes) y un secuenciador de 16 pistas.

El precio de estos equipos varía desde 250.000 a 500.000 ptas.

La alternativa a los Workstation es un teclado maestro y uno o varios módulos de sonidos. Los módulos de sonido son equipos que tienen unos sonidos básicos, una librería de formas de onda para elaborar sonidos y unos efectos (reverb, chorus, flanger etc.) para incorporar a dichos sonidos. La ventaja de los módulos de sonido es que ocupan muy poco espacio, y se pueden conectar unos con otros.

El precio de los módulos de sonido varia desde 90.000 hasta las 280.000 ptas.

Los teclados maestro son teclados que se conectan a los módulos de sonido. No tienen ningún sonido incorporado pero tienen una gran ventaja: hay teclados maestros de 4, 5, y de 6 octavas y media o más. (En los Workstation, los teclados que tienen incorporados suelen ser de 5 octavas)

El precio de los teclados maestros varía desde 25.000 ptas. (4 octavas) hasta 130.000 ptas. (7 octavas).

Normalmente las casas importantes como KORG y ROLAND suelen sacar al mercado un Workstation y aparte el modulo de sonidos equivalente, de manera que si una persona quiere los sonidos que tiene un Workstation y tiene un teclado maestro no tiene por qué comprarse todo el Workstation; le bastará con comprarse el módulo de sonidos equivalente por un precio que suele ser la mitad del precio del Workstation.

Realmente cuesta lo mismo comprarse un Workstation, que su módulo de sonidos equivalente y un teclado maestro. Y si luego queremos comprarnos otro módulo para ampliar nuestro equipo de sonidos, se puede conectar tanto al Workstation como al modulo de sonidos que tenemos, pero como dijimos anteriormente, trabajar directamente en el secuenciador del Workstation es más incómodo y menos intuitivo que si trabajamos

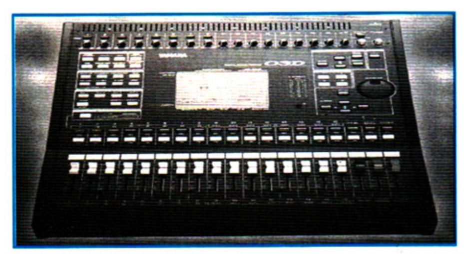
PRESUPUESTOS

A continuación tenéis 2 presupuestos aproximados. El primero de ellos se ciñe a las características de un estudio MIDI, mientras que el segundo está orientado hacia un estudio de grabación MIDI-AUDIO.

Presupuesto 1:
- Ordenador Pentium 100 o superior con 16 Megas
- Tarjeta de Sonido 32 bits o superior con 4 o 12 Megas
- Módulo de Sonidos (1 o más)
- Teclado maestro
- Secuenciador
- (Entre 200.000 ptas. y 400.000 ptas., aproximadamente)

Presupuesto 2:
- Ordenador Pentium 166MMX o superior con 2 gigas (o más) de disco duro y 32-64 Megas
- Tarjeta de Sonido 32 bits o superior con 4 o 12 Megas
- Módulo de Sonidos (1 o más)
- Teclado maestro
- Mesa de mezclas
- Secuenciador y editor de sonidos
- (Entre 400.000 y 600.000)

En estos presupuestos se puede sustituir el módulo y el teclado por un Workstation.

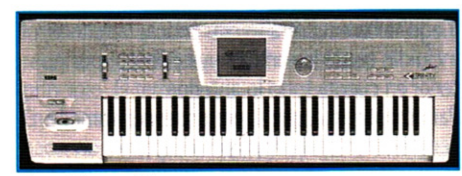

Mesa YAMAHA 03D.



Caja de Ritmos.



"El no va mas" JV-1080 de ROLAND.


Korg Trinity Workstation.

en el secuenciador de nuestro ordenador.
Otra de las ventajas importantes de comprarse por separado el módulo y el teclado es que si queremos comprarnos otro teclado podemos vender el que tenemos y comprarnos el otro teclado (igualmente pasa con los módulos), mientras que en el Workstation al venir todo incorporado en el mismo equipo, no podemos "sacar" el teclado del equipo y cambiarlo por otro teclado de más octavas (o de otras características); tendríamos que comprarnos todo un equipo nuevo para poder tener un teclado mejor (Aunque podríamos comprarnos un teclado y conectarlo al Workstation).

LA TARJETA DE SONIDO

La tarjeta de sonido es imprescindible en la elaboración de nuestro estudio de sonido. Tanto si se va a utilizar o no un Workstation, a la hora de grabar nuestra música en el disco duro del ordenador necesitaremos la tarjeta de sonido.
Tenemos que tener en cuenta que la tarjeta de sonido nos sirve de nexo entre el ordenador y el equipo musical del que disponemos.
Otra característica (muy interesante) de la tarjeta de sonido es que puede realizar la misma función que la de un módulo de sonidos. Actualmente las tarjetas disponen de memoria suficiente para albergar un gran número de sonidos.
A diferencia que los módulos, las tarjetas (de calidad semiprofesional) tienen la cualidad de poder introducir cualquier sonido analógico (como puede ser una voz) y convertirlo en sonido digital. Una tarjeta de sonido es como un pequeño sampler (por supuesto que está mucho más limitada que un sampler, pero viene a funcionar de la misma manera).
Por un precio aproximado a 40.000 ptas. podemos disponer de una tarjeta de sonido bastante interesante.
Si lo que quieres es investigar en esto de la música informática y de momento no quieres invertir mucho dinero, con un ordenador, una tarjeta de sonido y con un teclado maestro (si vas a realizar "música MIDI") tendrás suficiente para poder experimentar y crear los cimientos de tu estudio de sonido.

¿NECESITO MAS SONIDOS?

Uno de los problemas más comunes en la música sintética es el de disponer de un sonido concreto en el momento adecuado.
Si disponemos de una tarjeta de sonido con suficiente memoria, no es necesario que nos compremos otro módulo de sonidos. La amplia demanda de sonidos y de "riffs" de todos los estilos musicales (Rock, Jazz, Dance, Hip-Hop etc.) "ha conseguido" que en las tiendas especializadas en equipos de sonido profesional tengamos a nuestro alcance una gran cantidad de discos compactos con megas y más megas de esos sonidos que nuestros módulos o tarjetas no tienen. Esta opción de ampliar la librería de sonidos es muy interesante, ya que cada CD viene a costar en torno a las 5.000 ptas.

EL RESTO DEL EQUIPO

Dependiendo del equipo que tengamos y de nuestras necesidades, es importante tener en cuenta otros gastos. Si vamos a componer canciones con voz o efectos de sonido hablados, tendremos que comprarnos un micrófono. Por lo general, los micrófonos que vienen con las tarjetas de sonido son bastante "malillos". Si quieres comprarte un micrófono, piensa que a partir de unas 8.000 ptas. puedes encontrar uno aceptable.
También es importante saber que si vas a disponer de más de un módulo de sonido, tienes que amplificar cada módulo por separado. Pero si dispones de una mesa de mezclas, no tienes por qué preocuparte. El valor aproximado de una mesa de mezclas aceptable es de 70.000 ptas. A la hora de elegir una mesa de mezclas tenéis que tener en cuenta el número de canales. ecualización, reverb etc, dependiendo la elección de nuestras necesidades. Es importante no dejarse engañar por las apariencias. En el mercado existen muchos modelos diferentes de mesas, algunos muy atractivos para la vista y otros no tanto pero igualmente útiles y de muy buena calidad.

LA ELECCION CORRECTA

Antes de comprar un módulo de sonidos tenemos que tener bien claro qué tipo o estilo de música vamos a hacer. En el mercado existen módulos "genéricos" (tienen sonidos de bajos, baterías, guitarras, cuerdas, vientos etc.) de buena calidad rondando las 130.000 ptas.
También hay casas especializadas en instrumentos concretos, esto es, módulos de sonido que sólo tienen sonidos de bajos, módulos que sólo tienen sonidos de baterías, etc (la mayoría de estos módulos están orientados a música dance o techno). Estos módulos "individuales" vienen a costar en torno a 90.000 pesetas (o más) dependiendo del número de sonidos y su calidad.
Por último, existen módulos que tienen una base genérica y permiten ampliaciones por medio de tarjetas y placas de sonidos específicos. Por ejemplo una tarjeta (o placa) con sonidos étnicos, otra con sonidos de instrumentos de orquesta, otra con sonidos de bajo, etc. El precio de estos módulos ronda las 250.000 ptas. Y aunque en un principio sean caros, puede que sea la elección correcta, ya que se pueden ampliar (como los ordenadores) según necesitemos más recursos.

MOD, FT2 Y DEMAS

Apenas hemos comentado nada acerca de formatos MOD, S3M y FT2. Pues bien, los usuarios de trackers estáis de enhorabuena, ya que para poder hacer canciones en estos formatos, con una tarjeta de sonido, un ordenador 486 o superior y un CD para poder grabar sonidos en vuestro disco duro tenéis todo lo necesario para componer vuestras canciones.
La razón por la que en este artículo hemos dejado de lado este "tipo" de música es que en la elaboración de la banda sonora de un videojuego, este formato ha quedado anticuado. Pero no desesperéis; durante el mes que viene y los siguientes, hablaremos sobre los trackers, su utilización y diferentes trucos que os servirán de gran ayuda.

CONCLUSION

Claro está que la elección del equipo a comprar es exclusivamente vuestra, pero es importante que os asesoréis primero por entendidos en la materia.
Si empezáis de cero, no os preocupéis y compraros el equipo poco a poco; afortunadamente por muchos equipos nuevos que vayan saliendo en el mercado (esto no es el mundo de la informática) el equipo que compréis (si se ha elegido correctamente) tardará unos cuantos años en quedarse obsoleto.

Autor: Rafael S. Gago

Creación de una portada

Continuando con los trabajos realizados, en colaboración, entre grafistas 2D y grafistas 3D, esta vez nos centraremos en la creación de una hipotética portada que puede servir tanto para ser la portada principal de un juego, como para ser una de las imágenes que se visualizan en los momentos de la carga del juego.

En el número anterior de la revista, mostramos la manera en la cual una coordinación entre grafistas especializados en las 2 dimensiones y grafistas especializados en las 3 dimensiones, puede dar como resultado la creación de un terreno que será usado, posteriormente, por el grupo de programación o bien utilizarlo como una imagen de render para la realización de una imagen que sirva para ser usada en una portada. Pues bien, de esto precisamente es de lo que vamos a partir en este artículo, ya que la creación de una portada puede partir del terreno que anteriormente habíamos creado.

DE NUEVO LA COLABORACION

Una vez más vamos a hacer especial hincapié en lo fundamental que resulta la colaboración y la coordinación entre los antes mencionados grupos de grafistas, esto es 2D y 3D. La razón por la que dicha colaboración es esencial es porque la portada que vamos a realizar es un tipo de imagen que comenzará en un entorno de software tridimensional y que va a finalizar siendo retocada, una vez renderizada, con un programa de retoque fotográfico, gracias al cual, debido a los filtros que este tipo de software maneja, va a adquirir un aspecto final que sería muy trabajoso de conseguir si únicamente se usa un software de 3D.

LA IMPORTANCIA DE LA REALIZACION

Cuando llevamos a cabo la realización de una imagen que no va a ser retocada posteriormente con un programa al afecto, debemos preocuparnos especialmente por darle un aspecto lo más parecido a la realidad posible (siempre y cuando, evidentemente, la escena que pretendamos hacer sea un reflejo del mundo real, de lo contrario la imaginación es la única pauta a seguir). Para ello, es necesario que cuidemos de forma especial la modelación y la texturización de los objetos que van a formar parte de la escena, la iluminación con la que queramos ambientar el mundo en el que vamos a colocar los modelos y, en fin, todo lo necesario para que la imagen refleje de una manera fiel lo que hemos ideado y abocetado con anterioridad.

Ello nos llevará tiempo y una innumerable cantidad de pruebas hasta conseguir algo que empiece a gustarnos pero, una vez conseguido el resultado, no tenemos más que realizar el render de la malla y de los efectos que le hayamos añadido y dar por concluido el trabajo.

En cambio, cuando se prepara una imagen que se pretende que sea retocada por otra persona y con otro programa que no es el original de creación de la imagen, hay que contar con una serie de factores que, si bien por una parte facilitan bastante la labor de iluminación y presentación final, por otra, nos obligan a tener muy en cuenta todos y cada uno de los factores que van a formar parte de la imagen, y que, como vamos a ver a continuacióntienen una importancia decisiva que sólo se verá en el resultado final, una vez haya pasado por las manos de un grafista 2D.

Lo primero, como es natural en estos casos, es tener muy claro desde el principio lo que queremos que refleje la escena, para, posteriormente, poner en escena los elementos más adecuados al ambiente que queremos que posea.

Imaginemos que el motivo de la pantalla de presentación es un hipotético juego de guerra (el motivo de elegir este clase de juego no es otro que la vistosidad que se le puede dar a una imagen de este tipo, amén de que en el mercado se moverá con bastante soltura). Pues bien, si lo que queremos es que quede reflejado en un solo cuadro uno de los momentos más emocionantes del juego y que, a la vez, describa por sí mismo el tipo de juego que trata, evidentemente deberemos optar por escoger una imagen en la cual se produzca un ataque.

Ahora tendremos que pensar qué tipo de ataque podemos plasmar en nuestra portada. Para ello, deberemos tener en cuenta varios factores que pocas veces pasan por la cabeza de un grafista en circunstancias normales, pero que son de suma importancia cuando se orientan a la venta de un juego al público.

Nos referimos a que, si bien deberemos pensar en una imagen que refleje lo más perfectamente posible el tipo de juego, dicha imagen no debe tener contenidos demasiado explícitos, es decir, debemos conseguir que quien compre el juego sepa perfectamente y, sin lugar a dudas, que se trata de un juego con alto contenido de violencia pero, a la vez, la imagen que le ofrecemos y que va a ser la que recuerde cada vez que le hablen del juego, no sea en sí hiriente.

Pocas veces en un juego veremos que su portada tiene escenas explícitas de violencia, aunque luego el juego las posea. Y eso es, precisamente, lo que vamos a hacer. Por supuesto cabe la posibilidad de que escenas de alto contenido violento sean objeto de portada, pero deberemos tener en cuenta que eso puede ser motivo de problemas en focos de venta y distribución. No sería la primera vez que un juego se ha encontrado con críticas negativas, incluidas las de los críticos y el público, por no tener en cuenta todo lo anterior.

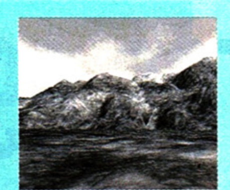
Para conseguir lo que se pretende, nos vamos a centrar en un ataque de un helicóptero a un tanque (todos sabemos que dentro de ambos hay personas, pero lo que se ve primeramente es que un aparato de metal va a quedar machacado, no que lo van a ser los que van dentro).

Pues bien, para ello comenzaremos modelando el tanque y el helicóptero. La forma de ambos objetos no tiene demasiada importancia salvo que se quiera hacer un juego con modelos reales de tanques o helicópteros. Basta, sencillamente, con hacer un modelo de tanque que se asemeje al concepto de dicho aparato que todos tendríamos en la cabeza si nos hablaran de alguna guerra determinada.

Una vez modelados ambos elementos pasaremos a texturizarlos. Este paso es bastante importante, puesto que los colores que lleven van a ser el estandarte del juego, y no es lo mismo si ponemos un tanque de color verde que si ponemos uno de tonalidades marrones, básicamente porque si la escena va tener una predominancia de colores ocres y rojos, un tanque de color verde (aunque pueda existir en la realidad) quedaría muy llamativo y quizá pareciera que pretendíamos llamar la atención hacia él por algún motivo. Como en este caso se verá, el tanque es, efectivamente, verde. La razón es que, en el juego, nosotros conducimos un tanque y tenemos que identificarnos más con él que con el helicóptero.




El comienzo es el modelado de los objetos.


El terreno es la segunda capa de la imagen.


Con el LenzFX le daremos viveza a la imagen.


La explosion le dara el toque final.


Este es el resultado del montaje de la imagen.

También, precisamente por ello, hemos optado por poner el tanque en primera línea, más cercano, de una manera similar a como lo veríamos en un juego poligonal. Y, por último, la razón de ladearlo ligeramente hacia la derecha no es otra que para darle una mayor sensación de volumen y de unión con el resto de la escena. Como se podrá comprobar, todas estas cosas en las que no nos fijamos cuando vemos una portada están cuidadosamente estudiadas. No conviene dejar nada al azar, puesto que esa imagen va a ser la descripción de nuestro juego.

UN COMIENZO EN TRES PARTES

Cuando ya tenemos terminados y texturizados los modelos que vamos a emplear, debemos retomar algo de lo que hemos hablado al principio. El terreno.
En el número anterior describimos la forma de hacer uno a nuestro antojo, y en este momento podemos echar mano de lo aprendido.
En este caso, el terreno (más bien la porción de terreno) tiene un primer plano liso. En él vamos a colocar nuestro tanque. Y tras la parte lisa del suelo, hemos tomado unas montañas. La razón para ello es doble: por una parte, no deja desnuda la línea del horizonte, cosa que queda bastante fea, y, por otra, nos brinda la oportunidad de hacer aparecer el helicóptero de las montañas, cosa que queda lo suficientemente aparente (no hay más que ver algunas películas norteamericanas, que usan este método para hacer que surja de la nada el malo de turno con una máquina lo suficientemente potente como para que nos corte por un momento el bocado de palomitas).
Cuando ya hemos situado ambos contendientes, cada uno en el lugar que les corresponde, (lo de poner el helicóptero a la derecha y el tanque a la izquierda tampoco es casual, tiene bastante que ver con el lugar por el que empezamos a escribir), llegamos a la parte más delicada del proceso: la iluminación. Y decimos la más delicada no porque no sepamos hacia dónde se deben dirigir las sombras de los objetos, sino porque si hay una lucha entre dos máquinas de este tipo es lógico pensar que habrá explosiones, y si hay explosiones hay una iluminación determinada por la luz que proviene de las mismas, y al fin y al cabo, esa luz es la que va a darle a la escena su personalidad.
Para ello hemos utilizado el IPAS conocido como LENZFX con el módulo INFERNO en su versión para 3D-Studio 4.
Para comenzar, hemos creado una serie de colores que simulen una explosión provocada por la caída de un misil, esto es, con unos colores amarillos y rojos muy vivos con un centro de explosión que tire a blanco.
Posteriormente, hemos utilizado el módulo INFERNO para generar un ruido fractal que pueda simular los gases y el fuego que dicho impacto provoca al tocar cualquier objeto.
Y una de las partes más importantes es el corte de la explosión. Es decir, cuando un misil cae en el suelo y explota, tanto la luz emitida por el fuego que provoca como el mismo fuego y los gases, se propagan hacia arriba, por ello, a nivel gráfico no podemos permitir que la explosión se expanda en este sentido, por lo tanto debemos tomar la línea del suelo como límite. Para esto, hemos usado los parámetros de opacidad lineal y opacidad radial hasta conseguir que la explosión quede cortada a izquierda y derecha.
Deberemos tener en cuenta que hay varios tipos de explosión, algunas explotan en todas direcciones y, por ello, el corte debe ser únicamente por la parte que correspondería al suelo; pero otras explotan básicamente hacia arriba y, en este caso, deberemos cuidar que el corte de dicha explosión sea mucho más grande, por lo que deberemos dejar únicamente un ángulo menor de 90º con su centro más o menos hacia arriba. Por supuesto que el corte no debe ser fuerte. Siempre hay que dejar que algo de ruido traspase las barreras del hipotético corte, porque de otra manera podía parecer que la explosión se está generando dentro de un gran embudo invisible, y eso le resta gran cantidad de realismo al efecto.
Hay una particularidad en lo referente a las explosiones, y viene dada por la fuga gaseosa que provocan los misiles que lanza el helicóptero hacia el tanque. Tales expulsiones de gases, fruto de la propulsión de estas armas, pueden ser solucionadas desde este módulo 3D o pueden ser dejadas en vacío para que sean los grafistas 2D los encargados de ponerlas en su sitio. En este caso, hemos escogido esta segunda opción, puesto que la realización de tal efecto con el LENZFX resulta más trabajoso y lleva más tiempo que si posteriormente se añade con un programa como bien puede ser nuestro viejo conocido PhotoShop.
Pues bien, una vez que tenemos todo esto terminado, tendremos que empezar la conocida labor de realizar una serie de render para que podamos hacernos una idea de si la iluminación de las explosiones se asemejan a la idea que teníamos preconcebida. Una vez que comprobemos que la escena se acerca a lo que pensábamos, es hora de preparar su paso a las 2D.
Para ello no hay ninguna dificultad. Una vez que tenemos hecho todo lo anterior, sólo habrá que dividirlo en tres partes. La razón no es otra que la de facilitar a los grafistas de 2D la labor de retoque. De esta manera, se evitarán mezclas en la aplicación de filtros y se simplificarán considerablemente las labores de retoque de sombras.
Lo único que tenemos que hacer es entregar la imagen en tres partes:
Por un lado tendremos el terreno. No es necesario que la iluminación sea la definitiva, puesto que luego va a ser retocado. Tendremos que tener gran cuidado en que se note que tiene volumen, es decir, si hay una montaña, la iluminación debe mostrárnosla, pues de lo contrario, la aplicación de filtros puede empobrecerla.
Por otro lado tendremos los modelos en tres dimensiones del tanque y el helicóptero. En este caso concreto deben tener suficiente nivel de detalle como para que puedan ser (especialmente el tanque) los objetos de la escena que van a soportar todo el peso gráfico y descriptivo.
Por último, tenemos las explosiones. Deben manejarse con cuidado, pues son las que tienen toda la carga en lo referente a la fuerza, al impacto que posee la imagen. No deben oscurecer ninguna parte de la escena y deben estar cargadas de movimiento.
Una vez entregadas estas tres imágenes a los grafistas 2D, tan sólo debemos esperar a que el retoque de imagen que ellos llevarán a cabo termine dando a la imagen la coherencia y la fuerza que se pretende.

Autor: Carlos Cabañas

Creación de fotos estáticas

En todo juego que se precie se tienen que cuidar todos los aspectos; no vale tener un juego gráficamente bueno sino que también la jugabilidad ha de encontrarse, como mínimo al mismo nivel. Por ello, la calificación de un juego se produce por un conjunto de elementos entre los que se encuentran los menús, las imágenes entre fase, las fotos, etc.

Como alguien dice *"la primera imagen es la que queda"* de ahí que estas primeras imágenes tengan que ser lo suficientemente atrayentes para que capte y "enganche " al jugador.
En el ejemplo práctico que proponemos este mes vamos a realizar una pantalla que podríamos situar en nuestro juego en el momento de carga del programa. La fotocomposición constaría de tres pasos claramente diferenciados, como son el montaje de los elementos tridimensionales que anteriormente nos pasaría el infografista, los efectos para resaltar la imagen y la creación del texto que coloquemos en la imagen.
El programa que utilizaremos en esta ocasión será el Photoshop 4.0 si bien son igualmente aceptables otros como el PhotoPaint o el Fractal Design.

MONTAJE DE LOS ELEMENTOS TRIDIMENSIONALES

El infografista nos pasará dos imágenes renderizadas: una con los elementos de la imagen (figura 1) y otra con el terreno (figura 2) donde se apoyaran los elementos 3D.
El montaje consistirá, por lo tanto, en situar los renderes de los objetos en el terreno. Estos objetos ya tienen creada la perspectiva con el 3D Max con lo que ganamos en tiempo ya que evitaremos tener que escalarlos.
Para realizar la fotocomposición sólo tenemos que abrirnos las dos imágenes.
Nos situamos sobre la imagen de los elementos; es importante que ésta sea de extensión TGA ya que el 3D Max, al lanzar el render, puede crear un canal alfa que nos será muy útil para recortar los objetos ya que detecta perfectamente la forma exacta de los elementos de la imagen.
Seleccionamos este canal alfa de la siguiente manera. Nos dirigimos al menú "Select", "Load selection" y pinchamos sobre la lista de los canales, seleccionando y activando el canal alfa, aceptamos y nuestros objetos habrán quedado perfectamente seleccionados.

El programa que utilizaremos en esta ocasión será el Photoshop 4.0

El siguiente paso es dejar los objetos en una nueva capa para tratarlos de manera independiente (menú "layer", "new", "layer by copy").
Lo que a continuación vamos a realizar es crearnos una capa (menú "layer", "new", "layer") y la situaremos debajo de los elementos 3D, pinchando sobre el "layer" recién creado en la ventana "layer", y Arrastramos hasta colocarla debajo de la de los elementos 3d.
En esta capa nos crearemos un efecto de humo para dotar a la imagen de la atmósfera adecuada que aún no tenemos. Rellenamos esta capa de cualquier color, después seleccionamos dos colores (uno para el background y otro para el foreground) dependiendo de los tonos que le queramos dar; una vez hecho esto le aplicamos el filtro "clouds" (menú "filters" "render" "clouds").
Para finalizar el primer paso sólo tendríamos que volcar las capas de los objetos recortados y la del humo a del terreno. Esto se hace con la herramienta de desplazamiento, pinchando sobre la capa correspondiente en la ventana de capas, mantener pulsado y volcarla sobre la otra imagen.

EFECTOS DE LA COMPOSICION

Ya en la imagen del terreno seleccionamos la capa de los elementos y ocultamos la del humo. Movemos nuestros objetos hasta que queden completamente acoplados en la orografía del terreno.
Para conseguir un efecto de perfecta armonía entre objetos y escenario ajustaremos los niveles de ambas capas para que ninguna resalte más de la otra; esto se consigue situándose en una de las capas (terreno o elementos) y pinchar sobre el menú "Image", "Adjust", "Levels" variando los niveles de luz hasta conseguir una perfecta compenetración tonal entre ambas capas. Lo siguiente sería proyectar sombras sobre el terreno; para lograrlo nos situaríamos en la

FIGURA 1.

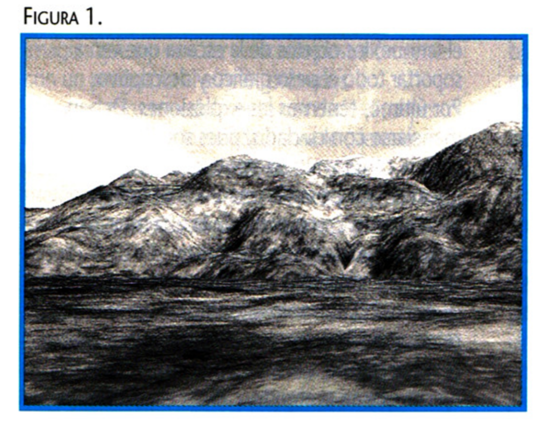


FIGURA 2.





FIGURA 3.



FIGURA 4.

capa del terreno y, por ejemplo, (hay multitud de maneras de proyección de sombras, vamos a utilizar la herramienta "Burn" para hacerlas). Nos situamos en la ventana "Options", movemos la presión de la herramienta hasta aproximadamente un 20% con la opción "Highlights" activada y seleccionamos una pluma con un grado de difuminado radial relativamente ancho. Ya sólo nos quedará seguir el contorno de, por ejemplo, las orugas de nuestro tanque para conseguir una sombra bastante aceptable.
El siguiente paso es terminar de crear el humo. Activamos la capa correspondiente y la dotamos de un cierto grado de transparencia y empezamos a modificar la forma. Hacemos una selección a mano alzada con un radio de calado de aproximadamente 15 píxeles ("Feather" en la ventana de opciones) y vamos eliminando zonas hasta conseguir la forma deseada.
Para dotar a la imagen de sensación de velocidad hemos aplicado el filtro "Radial blur" ("Filters", "Blur", "Radial blur") con la opción "zoom" activada . No utilizaremos un valor demasiado alto ya que esto nos distorsionaría demasiado la imagen y no sería posible distinguir los contornos del terreno.
Por último, sólo nos quedaría crear la iluminación utilizando unos "Flares" (menú "Render" "Lens flares") sin excedernos ya que podríamos llegar a saturar la imagen y quemarla.
A los objetos les iluminaríamos en zonas muy concretas con la herramienta "Dodge" con la opción "Highlights" activada, y con una pluma de difuminado lo suficientemente ancha y a un 12%, aproximadamente, de su capacidad ya que es una herramienta muy potente y clarea rápidamente en exceso la zonas en donde se aplica. Una vez hayamos terminado de aplicar los efectos sobre las diferentes capas, sólo resta juntar todas las capas (menú "Layer", "Flatten image"). Una vez hecho esto sólo nos queda crear el texto (figura 3) .

CREACION DE TEXTO

Para nuestro texto vamos a realizar un efecto de Photoshop que es el de letras deterioradas; para ello, tendremos que utilizar una serie de canales que nos servirán para crear las máscaras necesarias para modificar la imagen.
En primer lugar lo que tendríamos que hacer es abrir un nuevo canal (canal 4) en la ventana canales. Una vez creado lo rellenamos de blanco. Con el color del "Foreground" en blanco seleccionamos la herramienta de texto, con la opción "Feather" activada, y colocamos nuestro texto en el canal anteriormente creado. Lo siguiente es invertir el canal (menú "Image", "Adjust", "Invert") y, como resultado, nos tiene que quedar el texto en blanco con el fondo en negro. Nos creamos un nuevo canal (canal 5) e invertimos para que nos quede blanco, donde crearemos la selección que finalmente utilizaremos.

Para nuestro texto vamos a realizar un efecto de Photoshop que es el de letras deterioradas

Cargamos la selección del canal 4 (menú "Select", "Load selectión"). Para crear el texto deteriorado utilizaremos en primer momento el filtro "Noise" (menú "Filters", "Noise", "Add noise"), que modifica los niveles dependiendo de la cantidad de deterioro que se quiera para las letras.
En el siguiente paso a realizar tendremos que contornear en negro la selección; para ello, nos dirigiremos al menú "Edit", "Stroke" con un valor no demasiado alto y con la opción "Inside" activada para que la línea irregular que produzca nos salga de la selección hacia dentro.
Escoge el filtro "Diffuse" (menú "Stylize", "Diffuse"), selecciona la opción "Normal" y acepta. Deselecciona (memú "Select", "None") y aprieta Ctrl. F para repetir el filtro "Diffuse" con los mismos niveles. Usa después el filtro "Blur" (menú "Filter", "Blur") y aprieta Ctrl F tres veces para pronunciar el efecto de desenfoque.
Abre el menú "Image", "Levels" y mueve los reguladores blanco, negro y gris para obtener unas letras oscuras. Acepta para aplicar los cambios.
Si quieres una imagen sólo en blanco y negro, abre el menú "Image" y acciona el comando "Threshol" para convertir todos los píxeles de la imagen en blancos o negros.
Arrastra hacia la derecha si quieres conseguir zonas más negras o bien hacia la izquierda si pretendes lograr zonas más blancas.
Para terminar de trabajar con los canales invertimos el canal (menú "Image", "Adjust", "Invert") para que puedas activarlo en el canal RGB.
Activa el canal RGB y carga la selección del cana 5 (menú "Select", "Load selection"). Selecciona un color en la paleta de colores y rellena, si quieres, con un porcentaje de transparencia para que queden las letras un poco translúcidas. Si quieres puedes crearte una capa vía copia y rotar la selección para colocar el texto de diferente maneras y darle un poco de "vidilla" (figura 4) .

CREATIVIDAD SIN LIMITES

Autor: Daniel Pérez

DIV Games Studio

El mes pasado os informamos sobre una nueva compañía española llamada Hammer Technologies que está a punto de irrumpir en la escena. Esta compañía va a debutar con un impresionante proyecto llamado DIV Games Studio, que va salir a la venta dentro de muy poco tiempo. Verlo para creerlo. En la entrevista de este mes, hablamos con Daniel Navarro, el diseñador y programador principal de este innovador producto que promete crear escuela.

Parece no tener mucho sentido que una compañía que desarrolla videojuegos debute lanzando al mercado un entorno de desarrollo de videojuegos, ¿será algún truco? Nada de eso, cuando se ve DIV Games Studio por primera vez uno se queda abrumado por la cantidad de imágenes que le llegan a la cabeza; da la impresión de que se pueda estar durante las primeras 10 horas sin dejar de ver cosas nuevas e interesantes. Pero vayamos por partes; primeramente vamos a resumir de qué trata este producto sin igual.



La integracion entre herramientas es lo mejor de DIV.

DIV gestiona numerosas ventanas sin problemas.

DEFINICION

DIV es básicamente la única herramienta del mundo dentro de la cual se puede comenzar a crear un videojuego y terminarlo sin salir de ella. Se trata del primer lenguaje de programación profesional que ha sido diseñado específicamente para los videojuegos. ¿Un nuevo lenguaje?, sí, y mucho más. (Una última aclaración, olvidar cosas como Klik&Play; esto es infinitamente mejor, muchísimo más útil e increíblemente más completo.)

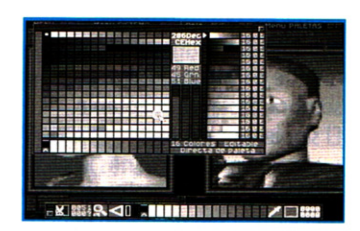
Una vez que te pones a analizarlo, lo primero que llama la atención al ver DIV es su atractivo entorno gráfico de ventanas, que se desplazan suavemente por él, cosa normal si pensamos que lo han hecho programadores de videojuegos.... Lo segundo es la cantidad de herramientas disponibles en el entorno, dentro de las cuales cabe destacar un intuitivo y potente editor de dibujo, con miles de opciones, botoncitos, parámetros, etc, y unas prestaciones que ya quisieran otros programas exclusivamente diseñados para esto. Lo tercero bien pudiera ser la enorme cantidad de videojuegos (más de 16) que se incluyen con DIV Games Studio. Todos ellos muy divertidos y con un acabado gráfico digno de profesionales; pero lo mejor es que ... ¡todos ellos están desarrollados íntegramente con DIV Games Studio!, y no sólo eso. También ¡aparecen con su código fuente!

Y cuando parece que ya lo has visto todo, entonces te encuentras con su editor de programas, con una ayuda en línea completísima, con la edición de animaciones, con un generador de tipos de letra, con otro generador de explosiones (¡¡), con un trazador gráfico de los programas, con la creación automática de instalaciones, con miles de gráficos y sonidos preparados para la creación de juegos, etc., ¡Ozú!, de verdad que se lo han *currao*.

EL MEJOR EN SU CAMPO

Con DIV no se pueden hacer simuladores de vuelo o juegos tipo *Quake*, pero esto es algo que sus creadores han prometido abordar en una próxima versión. No obstante, se pueden desarrollar con esta flexible herramienta la mayoría de los estilos de juego, esto es: estrategia, videoaventuras, tablero, lucha, simulaciones deportivas, plataformas, matamarcianos, juegos de rol (rpg), etc.

Si os queda alguna neurona en la cabeza, no dudaréis en ir al punto de venta más cercano y pagar 4.995 pesetas por esta pieza de relojería punta (¡mil duros por todo, con más de 16 juegos completos, y con sus listados!). De verdad, ¡os prometemos que no hay truco! Sólo esperamos que Hammer siga mucho, pero que mucho tiempo en activo ... y desde aquí les animamos a que continúen con la segunda versión de DIV, y con la tercera, y con la cuarta...







Hay muchas cosas que queremos saber, por ejemplo, ¿cómo ha salido de la nada una compañía española como Hammer?
¡No ha salido de la nada, jod...! Somos un gran grupo de profesionales que nos hemos dedicado a los videojuegos desde tiempos del Spectrum, Amstrad, etc, y que ahora nos hemos reunido para plantar cara, de una vez por todas, a tanto desarrollo "guiri" que invade nuestro país. Somos más de veinte programadores y grafistas, más la gente dedicada a las pelas (gestión, marketing y cosas así).

¿Cuánto tiempo ha durado el desarrollo de DIV Games Studio?
La idea maduró allá por el 92; se comenzó a programar la versión actual a principios del 95 y va a salir a la calle casi en el 98. Se comenzó con el diseño del nuevo lenguaje de programación, después se cambió, y entonces se volvió a cambiar. Estuvo cambiando casi hasta el final, pero creo que valió la pena tanto cambio. Después se programó el compilador, el editor gráfico, el entorno, etc. Siempre ha habido varias personas, aparte de mí, trabajando en el proyecto. Éstas han ido haciendo partes de la herramienta, los juegos ejemplo y demás, y también han tenido que trabajar unos cuantos (bastantes) grafistas durante este tiempo para completar todos los ejemplos.

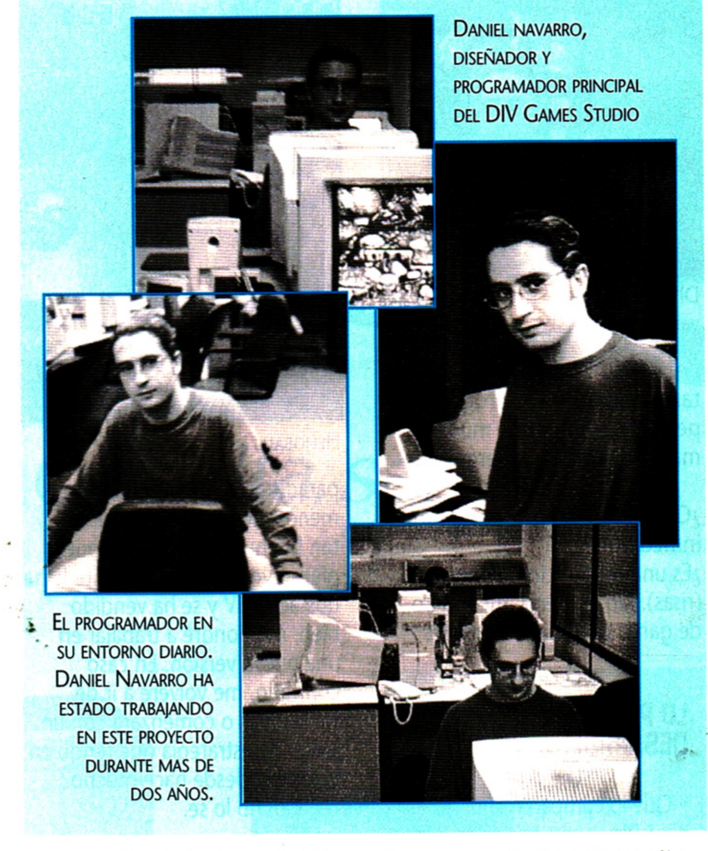
¿Cuál es el perfil de usuario al que va dirigido este producto?
Yo siempre he dicho que DIV podría ser utilizado por una persona que lo más parecido a un programa que hubiera visto en toda su vida fuera el Autoexec.bat. Y es cierto. DIV ha sido diseñado para que la gente aprenda a programar con él, y creo que esto es algo que resultará divertido para la gente, aunque yo tampoco soy imparcial, claro, pues soy un adicto a la programación desde bien chiquitito. Es algo parecido al dBase para hacer bases de datos, pero en juegos, que es mucho más fácil y potente que lenguajes de propósito general, como C.

¿Se pueden hacer realmente juegos comerciales con DIV? ¿Habéis pensado en hacer alguno vosotros?
Por supuesto que se puede. Sólo hace falta ponerse. Mucha gente me ha preguntado por qué no hemos trabajado un poco más alguno de los juegos-ejemplo de DIV y lo hemos vendido como juego comercial independiente; supongo que ha sido porque quería ofrecer todo lo posible junto a DIV. Intentar implantar a estas alturas un nuevo lenguaje de programación en el mercado es muy difícil, y sabíamos que teníamos que dar muchas facilidades para que la gente se acercara a él. Respecto a tu segunda pregunta te diré que sí, hemos comenzado a desarrollar un juego de estrategia comercial íntegramente desarrollado en DIV Games Studio.

¿Te molesta que te hablen de Klick&Play?
Sí, pues en mi modesta opinión, que de modesta no tiene nada, Klick&Play es sólo un juguete para niños. No he visto ningún desarrollo mínimamente profesional realizado con esta herramienta, ni con ninguna de sus continuaciones. Creo que la filosofía de DIV está más cerca

LO MEJOR DE ESTE DESARROLLO

- ¡Acabarlo de una vez por todas!, DIV Games Studio ha tenido una infancia más larga que la mía ... y diría que hasta más intensa.
- Haber podido hacer tantos videojuegos con él; me ha encantado versionar los clásicos y abordar tipos de juegos que no había realizado antes.
- Haber contando con tanto apoyo y sugerencias por parte de toda la gente que veía el producto (alguno incluso se permitió el lujo de hacer críticas).
- ¡La ilusión de que la gente pueda aprender a programar con DIV!, esta ha sido la esperanza que me ha dado fuerzas para acabarlo.
- Ha sido un desarrollo muy interesante por lo variado del mismo, no tiene nada que ver hacer el compilador, con los juegos, con el editor gráfico, con las librerías, etc...


Daniel Navarro, diseñador y programador principal del DIV Games Studio

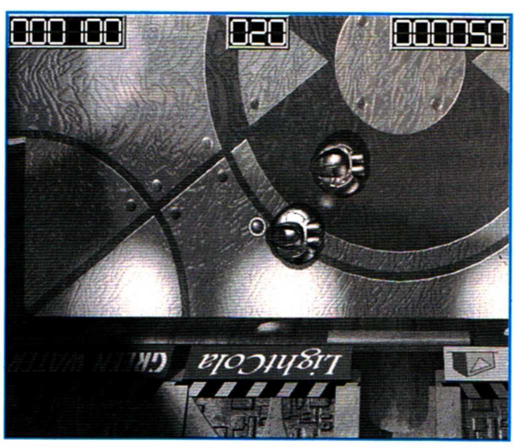
El programador en su entorno diario. Daniel Navarro ha estado trabajando en este proyecto durante mas de dos años.

CARACTERISTICAS DE LOS JUEGOS QUE SE PUEDEN DESARROLLAR EN DIV

- Cualquier resolución de vídeo (320x200 hasta 1024x768).
- Número ilimitado de sprites simultáneamente.
- Gestión de cualquier número de ventanas.
- Hasta 10 scrolles de varios planos simultáneamente.
- Control de límites de velocidad integrado.
- Manejo de planos abatidos en 3D (modo-7).
- Hasta 16 canales de sonido simultáneamente.
- Transparencias, rotaciones, escalados, ...
- Gestión automática de memoria.
- Música calidad CD-Audio.
- Control directo de dispositivos como ratón, joystick, teclado, ...
- Gestión de errores AAP avanzada (Amigable al programador).
- Sistema de capturas integrado.
- Detección perfecta de colisiones "al píxel".
- Múltiples efectos de paleta (fades, rotaciones, iluminación...).
- Autodetección de sistemas de sonidos (SoundBlaster y Gravis).
- Creación automática de setup de sonido, instalaciones, ...



DIV REVOLUCIONARA EL MUNDO DE LOS VIDEOJUEGOS



HELIOBALL.



SOCCER.

del AMOS de Amiga, de donde también partió ese programa, pero con un interfaz mucho más integrado y coherente.

¿Cuáles son tus planes de futuro inmediatos?
¿Es una proposición indecente? (risas). No, en serio, me muero de ganas por unas vacaciones paradisiacas en compañía de, al menos, dos mujeres que estén tan buenas como yo. Después pienso volver y, si a la gente le ha gustado DIV y se ha vendido bien, me pondré a trabajar en una nueva versión. En caso contrario, me volveré a ir de vacaciones o comenzaré con un juego de estrategia que tengo en la cabeza desde hace mucho tiempo, no lo sé.

¿Cuáles son tus juegos (y programas) favoritos?
Supongo que por programas te refieres a los de ordenador, ya que la caja tonta la utilizo sólo para ver películas de vídeo y jugar a la Nintendo 64. Te podría enumerar cientos de juegos que me han enamorado, desde el Manic-Miner, Bomb Jack, Green beret o The Sentinel, hasta el Super Mario 64, Wave Racer o Goldeneye, pasando por todos los intermedios. Como muy personales nombraría (si hay alguien que se acuerde de ellos) a XOR, El-Fish o Sim City. Como programas que me han impresionado, creo que el que más fue el primer emulador de ZX-Spectrum que ví, de Pedro Gimeno.

La pregunta del millón, ¿vas a hacer una versión de DIV para Windows 95? ¿y para hacer juegos 3D como *Quake*?
Vete a tomar por..., ahora que me comenzabas a caer bien. De acuerdo, por partes. El desarrollo para Windows'95 es muy costoso y para abordarlo necesitamos tener primero éxito con esta versión.



La filosofía de DIV es similar a la del AMOS de Amiga

Estaría bien que cada persona se comprara dos copias del programa. Y respecto a las 3D te diré que hemos pensado mucho y hay muchas ideas pinchadas en el corcho, si bien hemos de reconocer que no es el entorno ideal para aprender a programar. Es mejor hacerlo sobre las bases de 2D y después extrapolarlo; por ahora os tendréis que conformar con las prestaciones 3D actuales de DIV, aunque no permitan hacer "Quakes".

LO PEOR DE ESTE DESARROLLO:

• Que lo comparen con Klik&Play, ¡no tiene nada que ver!, DIV es muchísimo mejor y realmente es una herramienta profesional.

• Corregir los casi 3500 bugs de programación que surgieron durante el desarrollo ... ¡de verdad, creo que ya no queda ninguno!.

• Documentar páginas y páginas de manual para explicar todo lo que se podía hacer con DIV, fue algo eterno. ¡Y tener que traducir todo eso a un montón de idomas!

• No haber podido incluir todas las ideas que se nos han ocurrido en esta primera versión; esperemos que la gente comprenda que todavía no somos Microsoft.

• La incógnita de no saber cómo va a reaccionar la gente cuando salga DIV a la calle, ya que no hay precedentes de ningún programa parecido.

JUEGOS QUE VIENEN CON DIV GAMES STUDIO

FOSTIATOR	Lucha One Vs One, svga, parallax.
SPEED FOR DUMMIES	Carreras de coches, modo-7, split screen.
CHECK OUT	Tablero, inteligencia artificial, svga.
THE CASTLE OF DR. MALVADO	Plataformas, scroll de 3 planos multidireccional.
TOTAL BILLIARDS	Simulación física.
HELIOBALL	Futurista, svga, split screen, rotaciones.
WORLD CHAPAS CHAMP	Simulación arcade de habilidad.
SOCCER	Fútbol en 3D (vga/svga).
ALIEN SUPRIMER	Arcade de acción (tipo Raptor).
BLAST'EM UP	Arcade de acción (tipo Rtype).
PUZZLE'O MATIC	Arcade de habilidad y agudeza visual.

+ Versiones de clásicos PACOMAN (pacman), STEROID (asteroid), NOID (arkanoid), GALAX (galaxian), ...